Kernel Address Space Layout Randomization

http://outflux.net/slides/2013/lss/kaslr.pdf

Linux Security Summit, New Orleans 2013

Kees Cook <keescook@google.com>
(pronounced “Case”)

chrome

Overview

- Classic Attack Structure
- Address Space Layout Randomization
- Benefits
- Down-sides
- Useful Scenarios
- Implementation Details
- Demonstration
- Info Leaks

Classic Attack Structure

- Find arbitrary write bug
  - Endless stream of CVEs
- Insert malicious code into address space
  - Local userspace address? SMEP? Remote packet reception?
- Redirect execution flow
  - Return from function, close a socket, send a packet, whatever
- Run malicious code
  - commit_creds(prepare_creds())
- Clean up
  - Reset locks, fix overwritten structures, etc

Address Space Layout Randomization

- Disrupts finding where to write and execute
- Well established in userspace
  - Stack
  - Mmap (large heap, shared objects, “PIC”)
  - Brk (heap)
  - Text (“PIE”)
- Kernel ASLR has to start somewhere
  - Now: Text
  - Next: modules, kmalloc, vmalloc

Benefits

- IDT masked and read-only
- Statistical defense against attack
  - Target addresses are no longer fixed
- What happens when an attacker “misses”?
  - Userspace: daemon restarts...
    - Are you checking for repeated segfaults?
  - Kernel: entire system goes down
    - Are you checking for machine uptime?

Down-sides

- Hibernation
- Entropy
  - Source of randomness
  - Size of address space (2GiB in 2MiB chunks: max 1024)
- Secrecy
  - /proc/kallsyms (kptr_restrict)
  - dmesg (dmesg_restrict)
  - Log files (chmod)
  - Kernel objects exposed as API handles (e.g. INET_DIAG)

Useful Scenarios

- Local isolation
  - seccomp-bpf
  - namespaces
- Remote services
  - Many fewer leaks

Implementation Details

- git://git.kernel.org/pub/scm/linux/kernel/git/kees/linux.git
  - Branch “kaslr-c-v6”
  - Rolled out in Chrome OS
- Boot steps:
  - Figure out lowest safe address location
  - Walk E820 regions, counting kernel-sized slots
  - Choose slot randomly using best available method
    - RDRAND, RDTSC, or timer IO ports
  - Decompress, handle relocation, and start kernel
- Relocation support for 64-bit
- Expanded virtual memory layout of kernel image to 1GiB
- Panic message includes offset to aid debugging

chrome

Initial Boot Memory Layout

After boot loader...

0x0	BIOS and things
0x100000	Decompression code
...	Compressed kernel
...	Command line
...	Initrd
...	...empty...

Before decompression...

0x0	BIOS and things
0x100000	Decompression code
...	Stack, Heap
...	Command line
...	Initrd
...	...empty...
0x1000000	Target
...	
...	Compressed kernel
+ image size	...empty...

chrome

E820 Memory Regions

```
BIOS-e820: [mem 0x0000000000000000-0x0000000000000fff] type 16
BIOS-e820: [mem 0x0000000000001000-0x000000000009ffff] usable
BIOS-e820: [mem 0x00000000000a0000-0x00000000000fffff] reserved
BIOS-e820: [mem 0x0000000000100000-0x00000000eeffffff] usable
BIOS-e820: [mem 0x00000000f0000000-0x00000000ffffffff] reserved
BIOS-e820: [mem 0x0000000001000000-0x000000001fffffff] usable
BIOS-e820: [mem 0x0000000020000000-0x00000000201fffff] reserved
BIOS-e820: [mem 0x0000000020200000-0x000000003fffffff] usable
BIOS-e820: [mem 0x0000000040000000-0x00000000401fffff] reserved
BIOS-e820: [mem 0x0000000040200000-0x00000000acebffff] usable
BIOS-e820: [mem 0x00000000acec0000-0x00000000acffffff] type 16
BIOS-e820: [mem 0x00000000ad000000-0x00000000af9fffff] reserved
BIOS-e820: [mem 0x00000000f0000000-0x00000000f3ffffff] reserved
BIOS-e820: [mem 0x0000000100000000-0x000000014f5fffff] usable
```


chrome

Stock Virtual Memory Layout

0x0 - 0xffff800000000000		Userspace
...		Fun things
0xffff888000000000 - 0xffffc90000000000		kmalloc
0xffffc90000000000 - 0xffffea0000000000		vmalloc
...		Other fun things
0xffffffff80000000 - 0xffffffffa0000000	512 MiB	Text (-2 GiB)
0xffffffffa0000000 - 0xfffffffff000000	1532 MiB	modules
0xfffffffff000000 - 0xffffffffffffffff	4 MiB	Fixed-location stuff

chrome

kASLR Virtual Memory Layout

0x0 - 0xffff800000000000		Userspace
...		Fun things
0xffff888000000000 - 0xffffc90000000000		kmalloc
0xffffc90000000000 - 0xffffea0000000000		vmalloc
...		Other fun things
0xffffffff80000000 - 0xfffffffc0000000	1024 MiB	Text (-2 GiB)
0xfffffffc0000000 - 0xffffffffff000000	1020 MiB	modules
0xffffffffff000000 - 0xffffffffffffffff	4 MiB	Fixed-location stuff

chrome

Demonstration

- x86_64 .config contents
  - # CONFIG_HIBERNATION is not set
  - CONFIG_RELOCATABLE=y
  - CONFIG_RANDOMIZE_BASE=y
  - CONFIG_RANDOMIZE_BASE_MAX_OFFSET=0x40000000
  - CONFIG_PHYSICAL_ALIGN=0x200000
- Compare contents of
  - /proc/kallsyms
  - /sys/kernel/debug/kernel_page_tables (CONFIG_X86_PTDUMP)

chrome

Info Leaks

- Kernel addresses more valuable to attackers
  - Always use %pK
- Contents of dmesg needs to be protected
- Cannot use addresses as handles any more

chrome

Questions?

http://outflux.net/slides/2013/lss/kaslr.pdf

keescook@{chromium.org,google.com}

kees@outflux.net